EXPRESS MAIL LABEL NO. EV351283437US
DATE OF DEPOSIT: July 18, 2003

INTRAFRAME AND INTERFRAME INTERLACE CODING AND DECODING

RELATED APPLICATION INFORMATION

The following co-pending U.S. patent applications relate to the present application and are hereby incorporated herein by reference: 1) U.S. Patent Application Serial No. aa/bbb,ccc, entitled, "Advanced Bi-Directional Predictive Coding of Video Frames," filed concurrently herewith; and 2) U.S. Patent Application Serial No. aa/bbb,ccc, entitled, "Coding of Motion Vector Information," filed concurrently herewith.

COPYRIGHT AUTHORIZATION

TECHNICAL FIELD

Techniques and tools for interlace coding and decoding in interframes and intraframes are described. For example, a video encoder encodes macroblocks in an interlaced video frame in a 4:1:1 format.

BACKGROUND

Digital video consumes large amounts of storage and transmission capacity. A typical raw digital video sequence includes 15 or 30 frames per second. Each frame can include tens or hundreds of thousands of pixels (also called pels). Each pixel represents a tiny element of the picture. In raw form, a computer commonly represents a pixel with 24 bits. Thus, the number of bits per second, or bit rate, of a typical raw digital video sequence can be 5 million bits/second or more.

Most computers and computer networks lack the resources to process raw digital video. For this reason, engineers use compression (also called coding or encoding) to reduce the bit rate of digital video. Compression can be lossless, in which quality of the video does not suffer but decreases in bit rate are limited by the complexity of the video. Or, compression can be lossy, in which quality of the video suffers but decreases in bit rate are more dramatic. Decompression reverses compression.

In general, video compression techniques include intraframe compression and interframe compression. Intraframe compression techniques compress individual frames, typically called I-frames or key frames. Interframe compression techniques compress frames with reference to preceding and/or following frames, which are typically called predicted frames, P-frames, or B-frames.

Microsoft Corporation's Windows Media Video, Version 8 ["WMV8"] includes a video encoder and a video decoder. The WMV8 encoder uses intraframe and interframe compression, and the WMV8 decoder uses intraframe and interframe decompression.

A. Intraframe Compression in WMV8

Figure 1 illustrates block-based intraframe compression 100 of a block 105 of pixels in a key frame in the WMV8 encoder. A block is a set of pixels, for example, an 8x8 arrangement of pixels. The WMV8 encoder splits a key video frame into 8x8 blocks of pixels and applies an 8x8 Discrete Cosine Transform ["DCT"] 110 to individual blocks such as the block 105. A DCT is a type of frequency transform that converts the 8x8 block of pixels (spatial information) into an 8x8 block of DCT coefficients 115, which are frequency information. The DCT operation itself is lossless or nearly lossless. Compared to the original pixel values, however, the DCT coefficients are more efficient for the encoder to compress since most of the significant information is concentrated in low frequency coefficients (conventionally, the upper left of the block 115) and many of the high frequency coefficients (conventionally, the lower right of the block 115) have values of zero or close to zero.

The encoder then quantizes 120 the DCT coefficients, resulting in an 8x8 block of quantized DCT coefficients 125. For example, the encoder applies a uniform, scalar quantization step size to each coefficient. Quantization is lossy. Since low frequency DCT coefficients tend to have higher values, quantization results in loss of precision but not complete loss of the information for the coefficients. On the other hand, since high frequency DCT coefficients tend to have values of zero or close to zero, quantization of the high frequency coefficients typically results in contiguous regions of zero values. In addition, in some cases high frequency DCT coefficients are quantized more coarsely than low frequency DCT coefficients, resulting in greater loss of precision/information for the high frequency DCT coefficients.

The encoder then prepares the 8x8 block of quantized DCT coefficients 125 for entropy encoding, which is a form of lossless compression. The exact type of entropy encoding can vary depending on whether a coefficient is a DC coefficient (lowest frequency), an AC coefficient (other frequencies) in the top row or left column, or another AC coefficient.

The encoder encodes the DC coefficient 126 as a differential from the DC coefficient 136 of a neighboring 8x8 block, which is a previously encoded neighbor (e.g., top or left) of the block being encoded. (Figure 1 shows a neighbor block 135 that is situated to the left of the block being encoded in the frame.) The encoder entropy encodes 140 the differential.

The entropy encoder can encode the left column or top row of AC coefficients as a differential from a corresponding column or row of the neighboring 8x8 block. Figure 1 shows the left column 127 of AC coefficients encoded as a differential 147 from the left column 137 of the neighboring (to the left) block 135. The differential coding increases the chance that the differential coefficients have zero values. The remaining AC coefficients are from the block 125 of quantized DCT coefficients.

The encoder scans 150 the 8x8 block 145 of predicted, quantized AC DCT coefficients into a one-dimensional array 155 and then entropy encodes the scanned AC coefficients using a variation of run length coding 160. The encoder selects an entropy code from one or more run/level/last tables 165 and outputs the entropy code.

B. Interframe Compression in WMV8

Interframe compression in the WMV8 encoder uses block-based motion compensated prediction coding followed by transform coding of the residual error. Figures 2 and 3 illustrate the block-based interframe compression for a predicted frame in the WMV8 encoder. In particular, Figure 2 illustrates motion estimation for a predicted frame 210 and Figure 3 illustrates compression of a prediction residual for a motion-estimated block of a predicted frame.

For example, the WMV8 encoder splits a predicted frame into 8x8 blocks of pixels. Groups of four 8x8 blocks form macroblocks. For each macroblock, a motion estimation process is performed. The motion estimation approximates the motion of the macroblock of pixels relative to a reference frame, for example, a previously coded, preceding frame. In Figure 2, the WMV8 encoder computes a motion vector for a macroblock 215 in the predicted frame 210. To compute the motion vector, the encoder searches in a search area 235 of a reference frame 230. Within the search area 235, the encoder compares the macroblock 215 from the predicted frame 210 to various candidate macroblocks in order to find a candidate macroblock that is a good match. After the encoder finds a good matching macroblock, the encoder outputs information specifying the motion vector (entropy coded) for the matching macroblock so the decoder can find the matching macroblock during decoding. When decoding the predicted frame 210 with motion compensation, a decoder uses the motion vector to compute a prediction macroblock for the macroblock 215 using information from the reference frame 230. The prediction for the macroblock 215 is rarely perfect, so the encoder usually encodes 8x8 blocks of pixel differences (also called the error or residual blocks) between the prediction macroblock and the macroblock 215 itself.

Figure 3 illustrates an example of computation and encoding of an error block 335 in the WMV8 encoder. The error block 335 is the difference between the predicted block 315 and the original current block 325. The encoder applies a DCT 340 to the error block 335, resulting in an 8x8 block 345 of coefficients. The encoder then quantizes 350 the DCT coefficients, resulting in an 8x8 block of quantized DCT coefficients 355. The quantization step size is adjustable. Quantization results in loss of precision, but not complete loss of the information for the coefficients.

The encoder then prepares the 8x8 block 355 of quantized DCT coefficients for entropy encoding. The encoder scans 360 the 8x8 block 355 into a one dimensional array 365 with 64 elements, such that coefficients are generally ordered from lowest frequency to highest frequency, which typically creates long runs of zero values.

The encoder entropy encodes the scanned coefficients using a variation of run length coding 370. The encoder selects an entropy code from one or more run/level/last tables 375 and outputs the entropy code.

Figure 4 shows an example of a corresponding decoding process 400 for an inter-coded block. Due to the quantization of the DCT coefficients, the reconstructed block 475 is not identical to the corresponding original block. The compression is lossy.

In summary of Figure 4, a decoder decodes (410, 420) entropy-coded information representing a prediction residual using variable length decoding 410 with one or more run/level/last tables 415 and run length decoding 420. The decoder inverse scans 430 a one-dimensional array 425 storing the entropy-decoded information into a two-dimensional block 435. The decoder inverse quantizes and inverse discrete cosine transforms (together, 440) the data, resulting in a reconstructed error block 445. In a separate motion compensation path, the decoder computes a predicted block 465 using motion vector information 455 for displacement from a reference frame. The decoder combines 470 the predicted block 465 with the reconstructed error block 445 to form the reconstructed block 475.

The amount of change between the original and reconstructed frame is termed the distortion and the number of bits required to code the frame is termed the rate for the frame. The amount of distortion is roughly inversely proportional to the rate. In other words, coding a frame with fewer bits (greater compression) will result in greater distortion, and vice versa.

C. Bi-directional Prediction

Bi-directionally coded images (e.g., B-frames) use two images from the source video as reference (or anchor) images. For example, referring to Figure 5, a B-frame 510 in a video sequence has a temporally previous reference frame 520 and a temporally future reference frame 530.

Some conventional encoders use five prediction modes (forward, backward, direct, interpolated and intra) to predict regions in a current B-frame. In intra mode, an encoder does not predict a macroblock from either reference image, and therefore calculates no motion vectors for the macroblock. In forward and backward modes, an encoder predicts a macroblock using either the previous or future reference frame, and therefore calculates one motion vector for the macroblock. In direct and interpolated modes, an encoder predicts a macroblock in a current frame using both reference frames. In interpolated mode, the encoder explicitly calculates two motion vectors for the macroblock. In direct mode, the encoder derives *implied* motion vectors by scaling the co-located motion vector in the future reference frame, and therefore does not explicitly calculate any motion vectors for the macroblock.

D. Interlace Coding

A typical interlaced video frame consists of two fields scanned at different times. For example, referring to Figure 6, an interlaced video frame 600 includes top field 610 and bottom field 620. Typically, the odd-numbered lines (top field) are scanned at one time (e.g., time *t*) and the even-numbered lines (bottom field) are scanned at a different (typically later) time (e.g., time *t + 1*). This arrangement can create jagged tooth-like features in regions of a frame where motion is present because the two fields are scanned at different times. On the other hand, in stationary regions, image structures in the frame may be preserved (i.e., the interlace artifacts visible in motion regions may not be visible in stationary regions).

E. Standards for Video Compression and Decompression

Aside from WMV8, several international standards relate to video compression and decompression. These standards include the Motion Picture Experts Group ["MPEG"] 1, 2, and 4 standards and the H.261, H.262, and H.263 standards from the International Telecommunication Union ["ITU"]. Like WMV8, these standards use a combination of intraframe and interframe compression. The MPEG 4 standard describes coding of macroblocks in 4:2:0 format using, for example, frame DCT coding, where each luminance block is composed of lines from two fields alternately, and field DCT coding, where each luminance block is composed of lines from only one of two fields.

Given the critical importance of video compression and decompression to digital video, it is not surprising that video compression and decompression are richly developed fields. Whatever the benefits of previous video compression and decompression techniques, however, they do not have the advantages of the following techniques and tools.

SUMMARY

In summary, the detailed description is directed to various techniques and tools for encoding and decoding video images (e.g., interlaced frames). The various techniques and tools can be used in combination or independently.

In one aspect, macroblocks (e.g., in an interlaced video image) in a 4:1:1 format are processed. The 4:1:1 macroblocks comprise four 8x8 luminance blocks and four 4x8 chrominance blocks. The processing (e.g., video encoding or decoding) includes intra-frame and inter-frame processing. The macroblocks can be frame-coded macroblocks, or field-coded macroblocks having a top field and a bottom field.

In another aspect, a video encoder classifies a macroblock in an interlaced video image as a field-coded macroblock with a top field and a bottom field. The encoder encodes the top field and the bottom field using either an intra-coding mode or an inter-coding mode for each field. The coding modes used for encoding the top and bottom fields are selected independently of one another.

In another aspect, a video encoder sends encoded blocks in field order for a first field (e.g., an inter-coded field) and a second field (e.g., an intra-coded field) in a field-coded macroblock. The acts of sending encoded blocks in field order facilitate encoding the first field and the second field independently from one another. Intra-coded fields can be encoded using DC/AC prediction.

In another aspect, a video decoder receives encoded blocks in field order for a first encoded field and a second encoded field in a field-coded macroblock, and decodes the encoded fields. Receiving encoded blocks in field order facilitates decoding the first and second encoded fields independently from one another.

In another aspect, a video decoder finds a DC differential for a current block in the intra-coded field, finds a DC predictor for the current block, and obtains a DC value for the current block by adding the DC predictor to the DC differential. The intra-coded field is decoded independently from the second field.

In another aspect, a video decoder finds a DC differential for a current block in an intra-coded field and selects a DC predictor from a group of candidate DC predictors. The group of candidate DC predictors comprises DC values from blocks (e.g., previously decoded blocks) adjacent to the current block (e.g., the top, top-left, or left adjacent blocks). A candidate DC predictor is considered missing if it is not intra-coded, or if it is outside a picture boundary. The selected DC predictor is a non-missing candidate DC predictor.

In another aspect, a video encoder performs DC prediction for a current block in an interlaced macroblock and selectively enables AC prediction blocks in the macroblock. When the AC prediction is enabled, AC coefficients can be selected for differential coding based on the selected DC predictor for the current block. AC prediction can be signaled in a bit stream (e.g., with flags indicating whether AC prediction is performed for all blocks in a frame macroblock, or whether AC prediction is performed for blocks in a field in a field macroblock).

In another aspect, a video encoder finds a motion vector for an inter-coded field in a macroblock and encodes the macroblock using the motion vector for the first field, where the second field in the macroblock is an intra-coded field.

In another aspect, a video encoder finds a motion vector predictor for predicting a motion vector for a first field from among a group of candidate predictors. The candidate predictors are motion vectors for neighboring macroblocks, and the motion vector predictor is a motion vector for one corresponding field in a neighboring field-coded macroblock comprising two fields. The encoder calculates a motion vector for the first field using the motion vector predictor, and encodes the macroblock using the calculated motion vector. For example, the first field is a top field, and the one corresponding field in the neighboring field-coded macroblock is a top field.

In another aspect, a 4:1:1 macroblock in an interlaced video image is processed (e.g., in an encoder or decoder) by finding a luminance motion vector for the macroblock and deriving a chrominance motion vector for the macroblock from the luminance motion vector. The deriving can include scaling down the luminance motion vector by a factor of four. The chrominance motion vector can be rounded (e.g., to quarter-pixel resolution) and can be pulled back if it references an out-of-frame region in a reference frame.

In another aspect, a video decoder decodes a motion vector for a current interlaced macroblock (e.g., a frame or field macroblock) and obtains a prediction macroblock for the current macroblock using the decoded motion vector. The obtaining includes performing bi-cubic interpolation to obtain sub-pixel displacement for the current macroblock.

In another aspect, a 4:1:1 macroblock in a bi-directionally predicted video image (e.g., an interlaced image) is processed. The macroblock can be frame-coded macroblock (having up to two associated motion vectors) or field-coded (having up to four associated motion vectors). Direct mode macroblocks can also be classified as frame-type or field-type macroblocks.

Additional features and advantages will be made apparent from the following detailed description of different embodiments that proceeds with reference to the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a diagram showing block-based intraframe compression of an 8x8 block of pixels according to the prior art.

Figure 2 is a diagram showing motion estimation in a video encoder according to the prior art.

Figure 3 is a diagram showing block-based interframe compression for an 8x8 block of prediction residuals in a video encoder according to the prior art.

Figure 4 is a diagram showing block-based interframe decompression for an 8x8 block of prediction residuals in a video encoder according to the prior art.

Figure 5 is a diagram showing a B-frame with past and future reference frames according to the prior art.

Figure 6 is a diagram showing an interlaced video frame according to the prior art.

Figure 7 is a block diagram of a suitable computing environment in which several described embodiments may be implemented.

Figure 8 is a block diagram of a generalized video encoder system used in several described embodiments.

Figure 9 is a block diagram of a generalized video decoder system used in several described embodiments.

Figure 10 is a diagram showing luminance and chrominance samples in a 4:1:1 macroblock.

Figure 11 is a diagram showing an interlaced 4:1:1 macroblock.

Figure 12 is a diagram showing an interlaced 4:1:1 macroblock rearranged according to a field structure.

Figure 13 is a diagram showing an interlaced 4:1:1 macroblock subdivided into four 8x8 Y blocks, two 4x8 U blocks, and two 4x8 V blocks.

Figure 14 is a flow chart showing a technique for encoding fields in a field macroblock independently from one another.

Figure 15 is a diagram showing a technique for encoding 8x8 luminance blocks in a 4:1:1 macroblock.

Figure 16 is a diagram showing a technique for encoding 4x8 chrominance blocks in a 4:1:1 macroblock.

Figure 17 is a diagram showing a technique for decoding 8x8 luminance blocks in a 4:1:1 macroblock.

Figure 18 is a diagram showing a technique for decoding 4x8 chrominance blocks in a 4:1:1 macroblock.

Figure 19 is a diagram showing predictors for finding a DC coefficient for a current block.

Figures 20A and 20B are diagrams showing predictors for finding a motion vector for a frame-coded macroblock.

Figures 21A and 21B are diagrams showing predictors for finding one or more motion vectors for a field-coded macroblock.

DETAILED DESCRIPTION

The present application relates to techniques and tools for efficient compression and decompression of interlaced video. In various described embodiments, a video encoder and decoder incorporate techniques for encoding and decoding interlaced video frames, and signaling techniques for use in a bit stream format or syntax comprising different layers or levels (e.g., sequence level, frame/picture/image level, macroblock level, and/or block level).

The various techniques and tools can be used in combination or independently. Different embodiments implement one or more of the described techniques and tools.

I. Computing Environment

Figure 7 illustrates a generalized example of a suitable computing environment 700 in which several of the described embodiments may be implemented. The computing environment 700 is not intended to suggest any limitation as to scope of use or functionality, as the techniques and tools may be implemented in diverse general-purpose or special-purpose computing environments.

With reference to Figure 7, the computing environment 700 includes at least one processing unit 710 and memory 720. In Figure 7, this most basic configuration 730 is included within a dashed line. The processing unit 710 executes computer-executable instructions and may be a real or a virtual processor. In a multi-processing system, multiple processing units execute computer-executable instructions to increase processing power. The memory 720 may be volatile memory (e.g., registers, cache, RAM), non-volatile memory (e.g., ROM, EEPROM, flash memory, etc.), or some combination of the two. The memory 720 stores software 780 implementing a video encoder or decoder.

A computing environment may have additional features. For example, the computing environment 700 includes storage 740, one or more input devices 750, one or more output devices 760, and one or more communication connections 770. An interconnection mechanism (not shown) such as a bus, controller, or network interconnects the components of the computing environment 700. Typically, operating system software (not shown) provides an operating environment for other software executing in the computing environment 700, and coordinates activities of the components of the computing environment 700.

The storage 740 may be removable or non-removable, and includes magnetic disks, magnetic tapes or cassettes, CD-ROMs, DVDs, or any other medium which can be used to store information and which can be accessed within the computing environment 700. The storage 740 stores instructions for the software 780 implementing the video encoder or decoder.

The input device(s) 750 may be a touch input device such as a keyboard, mouse, pen, or trackball, a voice input device, a scanning device, or another device that provides input to the computing environment 700. For audio or video encoding, the input device(s) 750 may be a sound card, video card, TV tuner card, or similar device that accepts audio or video input in analog or digital form, or a CD-ROM or CD-RW that reads audio or video samples into the computing environment 700. The output device(s) 760 may be a display, printer, speaker, CD-writer, or another device that provides output from the computing environment 700.

The communication connection(s) 770 enable communication over a communication medium to another computing entity. The communication medium conveys information such as computer-executable instructions, audio or video input or output, or other data in a modulated data signal. A modulated data signal is a signal that has one or more of its characteristics set or changed in such a manner as to encode information in the signal. By way of example, and not limitation, communication media include wired or wireless techniques implemented with an electrical, optical, RF, infrared, acoustic, or other carrier.

The techniques and tools can be described in the general context of computer-readable media. Computer-readable media are any available media that can be accessed within a computing environment. By way of example, and not limitation, with the computing environment 700, computer-readable media include memory 720, storage 740, communication media, and combinations of any of the above.

The techniques and tools can be described in the general context of computer-executable instructions, such as those included in program modules, being executed in a computing environment on a target real or virtual processor. Generally, program modules include routines, programs, libraries, objects, classes, components, data structures, etc. that perform particular tasks or implement particular abstract data types. The functionality of the program modules may be combined or split between program modules as desired in various embodiments. Computer-executable instructions for program modules may be executed within a local or distributed computing environment.

For the sake of presentation, the detailed description uses terms like “indicate,” “choose,” “obtain,” and “apply” to describe computer operations in a computing environment. These terms are high-level abstractions for operations performed by a computer, and should not be confused with acts performed by a human being. The actual computer operations corresponding to these terms vary depending on implementation.

II. Generalized Video Encoder and Decoder

Figure 8 is a block diagram of a generalized video encoder 800 and Figure 9 is a block diagram of a generalized video decoder 900.

The relationships shown between modules within the encoder and decoder indicate the main flow of information in the encoder and decoder; other relationships are not shown for the sake of simplicity. In particular, Figures 8 and 9 generally do not show side information indicating the encoder settings, modes, tables, etc. used for a video sequence, frame, macroblock, block, etc. Such side information is sent in the output bit stream, typically after entropy encoding of the side information. The format of the output bit stream can be a Windows Media Video format or another format.

The encoder 800 and decoder 900 are block-based and use a 4:1:1 macroblock format. Each macroblock includes four 8x8 luminance blocks and four 4x8 chrominance blocks. Further details regarding the 4:1:1 format are provided below. The encoder 800 and decoder 900 also can use a 4:2:0 macroblock format with each macroblock including four 8x8 luminance blocks (at times treated as one 16x16 macroblock) and two 8x8 chrominance blocks. Alternatively, the encoder 800 and decoder 900 are object-based, use a different macroblock or block format, or perform operations on sets of pixels of different size or configuration.

Depending on implementation and the type of compression desired, modules of the encoder or decoder can be added, omitted, split into multiple modules, combined with other modules, and/or replaced with like modules. In alternative embodiments, encoder or decoders with different modules and/or other configurations of modules perform one or more of the described techniques.

A. Video Encoder

Figure 8 is a block diagram of a general video encoder system 800. The encoder system 800 receives a sequence of video frames including a current frame 805, and produces compressed video information 895 as output. Particular embodiments of video encoders typically use a variation or supplemented version of the generalized encoder 800.

The encoder system 800 compresses predicted frames and key frames. For the sake of presentation, Figure 8 shows a path for key frames through the encoder system 800 and a path for predicted frames. Many of the components of the encoder system 800 are used for compressing both key frames and predicted frames. The exact operations performed by those components can vary depending on the type of information being compressed.

A predicted frame (also called P-frame, B-frame, or inter-coded frame) is represented in terms of prediction (or difference) from one or more reference (or anchor) frames. A prediction residual is the difference between what was predicted and the original frame. In contrast, a key frame (also called I-frame, intra-coded frame) is compressed without reference to other frames.

If the current frame 805 is a forward-predicted frame, a motion estimator 810 estimates motion of macroblocks or other sets of pixels of the current frame 805 with respect to a reference frame, which is the reconstructed previous frame 825 buffered in a frame store (e.g., frame store 820). If the current frame 805 is a bi-directionally-predicted frame (a B-frame), a motion estimator 810 estimates motion in the current frame 805 with respect to two reconstructed reference frames. Typically, a motion estimator estimates motion in a B-frame with respect to a temporally previous reference frame and a temporally future reference frame. Accordingly, the encoder system 800 can comprise separate stores 820 and 822 for backward and forward reference frames. For more information on bi-directionally predicted frames, see U.S. Patent Application Serial No. aa/bbb,ccc, entitled, “Advanced Bi-Directional Predictive Coding of Video Frames,” filed concurrently herewith.

The motion estimator 810 can estimate motion by pixel, ½ pixel, ¼ pixel, or other increments, and can switch the resolution of the motion estimation on a frame-by-frame basis or other basis. The resolution of the motion estimation can be the same or different horizontally and vertically. The motion estimator 810 outputs as side information motion information 815 such as motion vectors. A motion compensator 830 applies the motion information 815 to the reconstructed frame(s) 825 to form a motion-compensated current frame 835. The prediction is rarely perfect, however, and the difference between the motion-compensated current frame 835 and the original current frame 805 is the prediction residual 845. Alternatively, a motion estimator and motion compensator apply another type of motion estimation/compensation.

A frequency transformer 860 converts the spatial domain video information into frequency domain (i.e., spectral) data. For block-based video frames, the frequency transformer 860 applies a discrete cosine transform ["DCT"] or variant of DCT to blocks of the pixel data or prediction residual data, producing blocks of DCT coefficients. Alternatively, the frequency transformer 860 applies another conventional frequency transform such as a Fourier transform or uses wavelet or subband analysis. If the encoder uses spatial extrapolation (not shown in Figure 8) to encode blocks of key frames, the frequency transformer 860 can apply a re-oriented frequency transform such as a skewed DCT to blocks of prediction residuals for the key frame. In some embodiments, the frequency transformer 860 applies an 8x8, 8x4, 4x8, or other size frequency transforms (e.g., DCT) to prediction residuals for predicted frames.

A quantizer 870 then quantizes the blocks of spectral data coefficients. The quantizer applies uniform, scalar quantization to the spectral data with a step-size that varies on a frame-by-frame basis or other basis. Alternatively, the quantizer applies another type of quantization to the spectral data coefficients, for example, a non-uniform, vector, or non-adaptive quantization, or directly quantizes spatial domain data in an encoder system that does not use frequency transformations. In addition to adaptive quantization, the encoder 800 can use frame dropping, adaptive filtering, or other techniques for rate control.

If a given macroblock in a predicted frame has no information of certain types (e.g., no motion information for the macroblock and no residual information), the encoder 800 may encode the macroblock as a skipped macroblock. If so, the encoder signals the skipped macroblock in the output bit stream of compressed video information 895.

When a reconstructed current frame is needed for subsequent motion estimation/compensation, an inverse quantizer 876 performs inverse quantization on the quantized spectral data coefficients. An inverse frequency transformer 866 then performs the inverse of the operations of the frequency transformer 860, producing a reconstructed prediction residual (for a predicted frame) or a reconstructed key frame. If the current frame 805 was a key frame, the reconstructed key frame is taken as the reconstructed current frame (not shown). If the current frame 805 was a predicted frame, the reconstructed prediction residual is added to the motion-compensated current frame 835 to form the reconstructed current frame. A frame store (e.g., frame store 820) buffers the reconstructed current frame for use in predicting another frame. In some embodiments, the encoder applies a deblocking filter to the reconstructed frame to adaptively smooth discontinuities in the blocks of the frame.

The entropy coder 880 compresses the output of the quantizer 870 as well as certain side information (e.g., motion information 815, spatial extrapolation modes, quantization step size). Typical entropy coding techniques include arithmetic coding, differential coding, Huffman coding, run length coding, LZ coding, dictionary coding, and combinations of the above. The entropy coder 880 typically uses different coding techniques for different kinds of information (e.g., DC coefficients, AC coefficients, different kinds of side information), and can choose from among multiple code tables within a particular coding technique.

The entropy coder 880 puts compressed video information 895 in the buffer 890. A buffer level indicator is fed back to bit rate adaptive modules.

The compressed video information 895 is depleted from the buffer 890 at a constant or relatively constant bit rate and stored for subsequent streaming at that bit rate. Therefore, the level of the buffer 890 is primarily a function of the entropy of the filtered, quantized video information, which affects the efficiency of the entropy coding. Alternatively, the encoder system 800 streams compressed video information immediately following compression, and the level of the buffer 890 also depends on the rate at which information is depleted from the buffer 890 for transmission.

Before or after the buffer 890, the compressed video information 895 can be channel coded for transmission over the network. The channel coding can apply error detection and correction data to the compressed video information 895.

B. Video Decoder

Figure 9 is a block diagram of a general video decoder system 900. The decoder system 900 receives information 995 for a compressed sequence of video frames and produces output including a reconstructed frame 905. Particular embodiments of video decoders typically use a variation or supplemented version of the generalized decoder 900.

The decoder system 900 decompresses predicted frames and key frames. For the sake of presentation, Figure 9 shows a path for key frames through the decoder system 900 and a path for predicted frames. Many of the components of the decoder system 900 are used for decompressing both key frames and predicted frames. The exact operations performed by those components can vary depending on the type of information being decompressed.

A buffer 990 receives the information 995 for the compressed video sequence and makes the received information available to the entropy decoder 980. The buffer 990 typically receives the information at a rate that is fairly constant over time, and includes a jitter buffer to smooth short-term variations in bandwidth or transmission. The buffer 990 can include a playback buffer and other buffers as well. Alternatively, the buffer 990 receives information at a varying rate. Before or after the buffer 990, the compressed video information can be channel decoded and processed for error detection and correction.

The entropy decoder 980 entropy decodes entropy-coded quantized data as well as entropy-coded side information (e.g., motion information 915, spatial extrapolation modes, quantization step size), typically applying the inverse of the entropy encoding performed in the encoder. Entropy decoding techniques include arithmetic decoding, differential decoding, Huffman decoding, run length decoding, LZ decoding, dictionary decoding, and combinations of the above. The entropy decoder 980 frequently uses different decoding techniques for different kinds of information (e.g., DC coefficients, AC coefficients, different kinds of side information), and can choose from among multiple code tables within a particular decoding technique.

A motion compensator 930 applies motion information 915 to one or more reference frames 925 to form a prediction 935 of the frame 905 being reconstructed. For example, the motion compensator 930 uses a macroblock motion vector to find a macroblock in a reference frame 925. A frame buffer (e.g., frame buffer 920) stores previously reconstructed frames for use as reference frames. Typically, B-frames have more than one reference frame (e.g., a temporally previous reference frame and a temporally future reference frame). Accordingly, the decoder system 900 can comprise separate frame buffers 920 and 922 for backward and forward reference frames.

The motion compensator 930 can compensate for motion at pixel, ½ pixel, ¼ pixel, or other increments, and can switch the resolution of the motion compensation on a frame-by-frame basis or other basis. The resolution of the motion compensation can be the same or different horizontally and vertically. Alternatively, a motion compensator applies another type of motion compensation. The prediction by the motion compensator is rarely perfect, so the decoder 900 also reconstructs prediction residuals.

When the decoder needs a reconstructed frame for subsequent motion compensation, a frame buffer (e.g., frame buffer 920) buffers the reconstructed frame for use in predicting another frame. In some embodiments, the decoder applies a deblocking filter to the reconstructed frame to adaptively smooth discontinuities in the blocks of the frame.

An inverse quantizer 970 inverse quantizes entropy-decoded data. In general, the inverse quantizer applies uniform, scalar inverse quantization to the entropy-decoded data with a step-size that varies on a frame-by-frame basis or other basis. Alternatively, the inverse quantizer applies another type of inverse quantization to the data, for example, a non-uniform, vector, or non-adaptive quantization, or directly inverse quantizes spatial domain data in a decoder system that does not use inverse frequency transformations.

An inverse frequency transformer 960 converts the quantized, frequency domain data into spatial domain video information. For block-based video frames, the inverse frequency transformer 960 applies an inverse DCT ["IDCT"] or variant of IDCT to blocks of the DCT coefficients, producing pixel data or prediction residual data for key frames or predicted frames, respectively. Alternatively, the frequency transformer 960 applies another conventional inverse frequency transform such as a Fourier transform or uses wavelet or subband synthesis. If the decoder uses spatial extrapolation (not shown in Figure 9) to decode blocks of key frames, the inverse frequency transformer 960 can apply a re-oriented inverse frequency transform such as a skewed IDCT to blocks of prediction residuals for the key frame. In some embodiments, the inverse frequency transformer 960 applies an 8x8, 8x4, 4x8, or other size inverse frequency transforms (e.g., IDCT) to prediction residuals for predicted frames.

When a skipped macroblock is signaled in the bit stream of information 995 for a compressed sequence of video frames, the decoder 900 reconstructs the skipped macroblock without using the information (e.g., motion information and/or residual information) normally included in the bit stream for non-skipped macroblocks.

III. Interlace Coding

Interlaced content (such as the interlaced content prevalent in the television industry) is an important consideration in video encoding and decoding applications. Accordingly, described embodiments include techniques and tools for efficient compression and decompression of interlaced video.

As explained above, a typical interlaced video frame consists of two fields (e.g., a top field and a bottom field) scanned at different times. Described embodiments exploit this property and perform efficient compression by selectively compressing different regions of the image using different techniques. Typically, it is more efficient to encode stationary regions as a whole (frame coding). On the other hand, it is often more efficient to code moving regions by fields (field coding). Therefore, in described embodiments, macroblocks in an image can be encoded either as frame macroblocks or field macroblocks. Frame macroblocks are typically more suitable for stationary regions. Field macroblocks are typically more suitable for moving regions because the two fields in the macroblock tend to have different motion, and each field tends to have a higher correlation with itself than with the other field. Some described embodiments focus on field macroblock encoding for both intra-coded frames and inter-coded frames.

The features of the described embodiments include:

1) A 4:1:1 YUV macroblock format for interframe and intraframe compression and decompression.

2) Inter-coding or intra-coding a field within a macroblock, independent of whether the other field within the macroblock was inter-coded or intra-coded.

3) A DC/AC prediction scheme that facilitates encoding fields independently of each other.

4) Motion vector prediction techniques for interlaced frames, including using motion vectors from neighboring fields separately to predict a motion vector for a current field, rather than averaging them.

5) A scheme for deriving chrominance motion vectors from luminance motion vectors.

A. 4:1:1 Macroblock Format

In some embodiments, a video encoder/decoder processes macroblocks in a 4:1:1 macroblock format. Figure 10 shows a 4:1:1 format for a macroblock. A 4:1:1 macroblock consists of a luminance matrix 1010 and two chrominance matrices 1020 and 1030. Relative to the luminance matrix, the chrominance matrices are sub-sampled by a factor of four in the horizontal dimension, but are at full resolution in the vertical dimension.

The 4:1:1 format differs from the 4:2:0 format in the arrangement of the chrominance samples. Both 4:1:1 and 4:2:0 macroblocks have four 8x8 luminance blocks. A 4:2:0 macroblock has two 8x8 chrominance blocks, one for each of the U and V channels. The U and V channels are therefore sub-sampled by a factor of two in both the vertical and horizontal dimensions. However, a 4:1:1 macroblock has four 4x16 chrominance blocks, two for each of the U and V channels. The 4:1:1 format preserves the field structure in the chrominance domain and has a better chrominance sub-sampling ratio, which results in accurate reconstruction of moving color regions in interlaced video.

Macroblocks in interlaced frames can be classified as frame macroblocks or field macroblocks. Figure 11 shows an original macroblock 1100 in 4:1:1 format. The original macroblock 1100 is composed of eight top field lines (odd-numbered lines 1, 3, 5, 7, 9, 11, 13 and 15) and eight bottom field lines (even-numbered lines 2, 4, 6, 8, 10, 12, 14 and 16). A frame macroblock has a layout identical to the original macroblock 1100. Figure 12 shows a field macroblock 1200. Field macroblock 1200 is rearranged relative to the original macroblock 1100, with the top field lines together in the top half and the bottom field lines together in the bottom half of the macroblock 1200.

As explained above, in interlaced frames, the top field lines and the bottom field lines are scanned at different times. Referring again to Figure 11, if the original macroblock 1100 contains fast moving objects, then the correlation among lines of the same field tends to be stronger than the correlation among lines of different fields (e.g., motion in line 1 has a stronger correlation with line 3 than with line 2, even though line 2 is closer to line 1 spatially). On the other hand, if the original macroblock 1100 contains mostly stationary objects, then the correlation among lines of different fields tends to be stronger than the correlation among lines of the same field (e.g., line 1 has a stronger correlation with line 2 than with line 3.) This is the reasoning behind classifying macroblocks as frame type or field type. For example, an encoder can select frame type for stationary to low-motion macroblocks and field type for high-motion macroblocks.

After a 4:1:1 macroblock is classified as a frame macroblock or a field macroblock, it is subdivided into blocks. For example, Figure 13 shows a macroblock 1300 subdivided into four 8x8 Y blocks (Y_0, Y_1, Y_2, Y_3), two 4x8 U (U_0, U_1) blocks and two 4x8 V (V_0, V_1) blocks. For a field macroblock, the top field comprises only blocks Y_0, Y_1, U_0, V_0, and the bottom field comprises only blocks Y_2, Y_3, U_1, V_1.

B. Independent Coding of Macroblock Fields

In some embodiments, one field in a field-coded macroblock is capable of being inter-coded or intra-coded regardless of how the other field in the macroblock was encoded. This allows the macroblock to contain one inter-coded field and one intra-coded field, rather than being restricted to being entirely intra-coded or inter-coded. This flexibility is helpful, for example, in scene transitions where the two fields of an interlaced frame are from different scenes. One field (e.g., a field in a macroblock corresponding to a newly introduced scene) can be intra-coded while the other field (e.g., a field corresponding to a previous scene) can be inter-coded (i.e., predicted from other frames).

For example, Figure 14 shows a technique 1400 for encoding fields in a field macroblock independently from one another. First, at 1410, an encoder classifies a macroblock as a field macroblock. Then, at 1420, the encoder encodes the top field in the macroblock using either intra-coding or inter-coding. At 1430, the encoder then encodes the bottom field using either intra-coding or inter-coding, regardless of whether the top field was intra-coded or inter-coded. Referring again to Figure 13, for a frame macroblock, the encoder sends the blocks in the following order: Y_0, Y_1, Y_2, Y_3, U_0, U_1, V_0, V_1. For a field macroblock, the encoder sends the blocks in field order: Y_0, Y_1, U_0, V_0 (top field) and Y_2, Y_3, U_1, V_1 (bottom field). For field macroblocks, the encoder sends the blocks in field order to allow intra- and inter-coded fields to exist within the same macroblock.

Finer encoding granularity (in terms of allowing for different kinds of motion in different fields) can be achieved when fields can be encoded independently from one another. To help achieve this finer granularity, some embodiments employ DC/AC prediction techniques for encoding an intra field independently from the other field in the macroblock.

1. DC/AC Prediction

In some embodiments, DC/AC prediction techniques facilitate the co-existence of inter- and intra-coded fields in the same macroblock. Figures 15 and 16 show exemplary techniques 1500 and 1600 for encoding macroblocks using DC/AC prediction techniques.

For example, when coding an interlaced video frame, an encoder encodes 4:1:1 macroblocks (which have been classified as either field macroblocks or frame macroblocks) in raster scan order from left to right. Referring again to Figure 13, a macroblock 1300 is subdivided into four 8x8 Y blocks (Y_0, Y_1, Y_2, Y_3), two 4x8 U (U_0, U_1) blocks and two 4x8 V (V_0, V_1) blocks. For a field macroblock, the top field comprises only blocks Y_0, Y_1, U_0, V_0, and the bottom field comprises only blocks Y_2, Y_3, U_1, V_1. An encoder encodes the blocks in different ways depending on, for example, whether a macroblock is a field or frame macroblock, and whether the block within the macroblock is a chrominance or luminance block.

Figure 15 shows a technique 1500 for encoding 8x8 luminance blocks (e.g., blocks Y_0, Y_1, Y_2, and Y_3 (Figure 13)). The encoder forms residual blocks for the luminance blocks. In some embodiments, the encoder forms residual blocks by subtracting an expected average pixel value from each pixel in the luminance blocks. For example, at 1510, the encoder subtracts 128 from each pixel (e.g., where the color depth ranges from 0 to 255) to form residual blocks. The encoder applies an 8x8 DCT 1520 to the residual blocks. The encoder performs DC/AC prediction along the row or the column of the residual blocks (e.g., residual 8x8 luminance block 1530). After DC/AC prediction, the encoder performs quantization 1540 on the coefficients, performs an 8x8 zig-zag scan 1550, and performs variable-length coding 1560 of the results.

Figure 16 shows a similar technique 1600 for encoding 4x8 chrominance blocks (e.g., blocks U_0, U_1, V_0, and V_1 (Figure 13)). The encoder forms residual blocks for the chrominance blocks (e.g., by subtracting 1610 a value of 128 from each pixel). The encoder applies a 4x8 DCT 1620 to the residual blocks. The encoder performs DC/AC prediction along the row or the column of the residual blocks (e.g., residual 4x8 chrominance block 1630). After DC/AC prediction, the encoder performs quantization 1640 on the coefficients, performs a 4x8 zig-zag scan 1650, and performs variable-length coding 1660 of the results.

For both the luminance and chrominance blocks, the encoder encodes DC coefficients differentially using the DC coefficients of neighboring blocks as predictors. While DC coefficients are always encoded differentially using neighboring blocks as predictors in these techniques, the encoder determines during encoding whether to predictively encode AC coefficients, and signals predictive AC coefficient encoding using flags (e.g., the ACPREDMB, ACPREDTFIELD, and/or ACPREDBFIELD flags described below). For a chrominance block, if row AC prediction is chosen, then the four coefficients of the first row are differentially coded.

Figures 17 and 18 show techniques 1700 and 1800 for decoding chrominance blocks and luminance blocks in 4:1:1 macroblocks. In Figure 17, at 1710, a decoder decodes variable length codes representing DC and AC coefficients in 8x8 luminance blocks. The decoder performs an inverse 8x8 zig-zag scan 1720 and performs DC/AC prediction for 8x8 luminance blocks (e.g., luminance block 1730). The decoder's completion of DC/AC prediction results in reconstructed, quantized, DCT luminance coefficient blocks. To complete the decoding, the decoder performs inverse quantization 1740 and an inverse DCT 1750 on the coefficients and adds 128 (at 1760) to each pixel.

In Figure 18, at 1810, a decoder decodes variable length codes representing DC and AC coefficients in 4x8 chrominance blocks. The decoder performs an inverse 4x8 zig-zag scan 1820 and performs DC/AC prediction for 4x8 chrominance blocks (e.g., chrominance block 1830). The decoder's completion of DC/AC prediction results in reconstructed, quantized, DCT chrominance coefficient blocks. To complete the decoding, the decoder performs inverse quantization 1840 and an inverse DCT 1850 on the coefficients and adds 128 (at 1860) to each pixel.

a. DC Prediction

In DC/AC prediction, the quantized DC value for the current block is obtained by adding the DC predictor to the DC differential. The DC predictor is obtained from one of the previously decoded adjacent blocks. For example, Figure 19 shows the current block 1910 and adjacent candidate predictor blocks. The values A, B and C in the adjacent candidate predictor blocks represent the quantized DC values (prior to the addition of 128) for the top-left, top and left adjacent blocks respectively.

In some cases, one or more of the adjacent candidate predictor blocks with values A, B, and C are considered missing. For example, a candidate predictor block is considered missing if it is outside the picture boundary. Or, when finding a predictor for a current intra block in an interlaced inter-frame (e.g., an interlaced P-frame), the candidate predictor block is considered missing if it is not intra-coded. Only values from non-missing predictor blocks are used for DC prediction.

In some embodiments, if all three candidate blocks are present, the encoder/decoder selects the predictor value based on the following rule:

```
If ( | B – A | < | C – A | ) {
    Predictor value = C
} else {
    Predictor value = A
}
```

If an adjacent candidate block is missing, then the following rules apply:

- If block C is missing and block B is not, then choose B as the predictor.
- If block B is missing and block C is not, then choose C as the predictor.
- If both B and C are missing, then no predictor is used.
- If A is missing, and B and C are present, then choose B if the DC predictor for block C is smaller than the DC predictor for block B, otherwise, choose block C.

Alternatively, an encoder/decoder uses other rules for choosing DC predictors.

b. AC Prediction

If AC prediction is enabled for the current block, then the AC coefficients on either the top row or the left column of the current block may be differentially encoded. This decision is based on the DC predictor. For example, in some embodiments, AC prediction proceeds according to the following rules:

- If the DC predictor is the top block, then the AC coefficients on the top row of the current block are differentially coded.
- If the DC predictor is the left block, then the AC coefficients on the left column of the current block are differentially coded.
- If no DC predictor is used, then the AC coefficients are not differentially coded.

Alternatively, an encoder/decoder uses other rules for AC prediction.

The AC coefficients in a predicted row or column are added to the corresponding decoded AC coefficients (prior to adding 128) in the current block to produce a reconstructed, quantized, DCT coefficient block.

2. Signaling for DC/AC Prediction

In some embodiments, an encoder/decoder uses signals in a bit stream at macroblock level to indicate whether AC prediction is active for a macroblock or for individual fields in a macroblock. For example, for frame macroblocks, an encoder indicates whether AC prediction will be performed for all blocks in the macroblock with the one-bit flag ACPREDMB. For field macroblocks, the encoder uses two one-bit flags to independently indicate whether AC prediction will be performed for blocks in the top field (ACPREDTFIELD) and bottom field (ACPREDBFIELD). Specifically, referring again to Figure 13, ACPREDMB indicates whether AC prediction is used for blocks Y_0, Y_1, Y_2, Y_3, U_0, U_1, V_0, and V_1 in a frame macroblock. In field macroblocks, ACPREDTFIELD indicates whether AC prediction is used for blocks Y_0, Y_1, U_0, and V_0, and ACPREDBFIELD indicates whether AC prediction is used for blocks Y_2, Y_3, U_1, and V_1. Alternatively, an encoder signals AC prediction in some other manner or at some other level.

C. Motion Vector Information in Inter-coded Interlaced Frames

As explained above, macroblocks are classified as frame macroblocks or field macroblocks and can be intra-coded or inter-coded. Thus, macroblocks can be one of four types: inter-coded frame macroblocks, inter-coded field macroblocks, intra-coded frame macroblocks, or intra-coded field macroblocks. Inter-coded macroblocks are motion compensated using motion vectors. For example, in P-frames, inter-coded frame macroblocks are motion compensated using one motion vector.

In some embodiments, inter-coded field macroblocks can have either one motion vector or two motion vectors. For example, when an inter-coded field macroblock has two motion vectors, each of the two fields in the macroblock has its own motion vector. On the other hand, when an inter-coded field macroblock has one motion vector, one of the two fields is intra-coded (not motion compensated) while the other field is inter-coded (motion compensated).

1. Motion Vector Predictors in Interlaced P-Frames

In general, motion vectors are computed by adding the motion vector differential to a motion vector predictor. In some embodiments, the motion vector predictor is computed using motion vectors from three neighboring macroblocks. For example, an encoder/decoder computes the motion vector predictor for a current macroblock by analyzing motion vector predictor candidates of the left, top, and top-right macroblocks. The motion vector predictor candidates are computed based on the current macroblock type.

Figures 20A and 20B show motion vector predictor candidates for frame macroblocks, and Figures 21A and 21B show motion vector predictor candidates for field macroblocks. For example, Figure 20A shows predictors for finding a motion vector for a current frame macroblock 2010 that is not the last macroblock in a macroblock row, while Figure 20B shows predictors for finding a motion vector where the current frame macroblock 2010 is the last macroblock in a macroblock row. The predictor candidates are computed differently depending on the whether the neighboring macroblock is frame-coded or field-coded. If the neighboring macroblock is frame-coded, its motion vector is taken as the predictor candidate. On the other hand, if the neighboring macroblock is field-coded, its top and bottom field motion vectors are averaged to form the predictor candidate.

Figure 21A shows predictors for finding one or more motion vectors for a current field macroblock 2110 that is not the last macroblock in a macroblock row, while Figure and 21B shows predictors for finding one or more motion vectors where the current field macroblock 2110 is the last macroblock in a macroblock row. Motion vectors for the corresponding fields of the neighboring macroblocks are used as predictor candidates. If a neighboring macroblock is field-coded, the predictor candidate for the top field is taken from the neighboring macroblock's top field, and the predictor candidate for the bottom field is taken from the neighboring macroblock's bottom field. When a neighboring macroblock is frame-coded, each of the motion vectors corresponding to its two fields are deemed to be equal to the motion vector for the macroblock as a whole. In other words, the top field and bottom field motion vectors are set to V, where V is the motion vector for the entire macroblock.

In both cases, if there are no motion vectors for the candidate neighboring field or macroblock (e.g., the field or macroblock is intra coded), the motion vector for the candidate neighboring field or macroblock is set to be zero.

The predictor is calculated by taking the component-wise median of the three candidate motion vectors. For more information on median-of-three prediction, see U.S. Patent Application Serial No. aa/bbb,ccc, entitled, "Coding of Motion Vector Information," filed concurrently herewith. Alternatively, the predictor is calculated using some other method.

2. Derivation of Chrominance Motion Vectors from Luminance Motion Vectors

In some embodiments, an encoder/decoder derives chrominance motion vectors from luminance motion vectors. For example, an encoder/decoder reconstructs a chrominance motion vector for a macroblock from the corresponding frame/field luminance motion vector. For frame-coded macroblocks, there will be one chrominance motion vector corresponding to the single luminance motion vector for the macroblock. On the other hand, for field-coded macroblocks, there will be two chrominance motion vectors corresponding to the two luminance motion vectors for the macroblock (e.g., one motion vector for the top field and one motion vector for the bottom field).

An encoder/decoder can use the same rules for deriving chrominance motion vectors for both field and frame macroblocks; the derivation is only dependent on the luminance motion vector, and not the type of macroblock. In some embodiments, chrominance motion vectors are derived according to the following pseudo-code:

```
frac_x4 = (lmv_x << 2) % 16;
int_x4 = (lmv_x << 2) – frac_x;
ChromaMvRound [16] = {0, 0, 0, .25, .25, .25, .5, .5, .5, .5, .5, .75, .75, .75, 1, 1};
cmv_y = lmv_y;
cmv_x = Sign (lmv_x) * (int_x4 >> 2) + ChromaMvRound [frac_x4];
```

cmv_x and cmv_y are chrominance motion vector components and lmv_x and lmv_y are corresponding luminance motion vector components. cmv_x is scaled by four while cmv_y is not scaled. The 4:1:1 format of the macroblock requires no scaling of in the y dimension. This derivation technique is therefore well-suited for a 4:1:1 macroblock format. The scaled cmv_x is also rounded to a quarter-pixel location. Rounding leads to lower implementation costs by favoring less complicated positions for interpolation (e.g., integer and half-integer locations).

After cmv_x and cmv_y are computed, the encoder/decoder can check to see if components should be pulled back (e.g., if the components map to an out-of-frame macroblock.) For more information on motion vector pull-back techniques, see U.S. Patent Application Serial No. aa/bbb,ccc, entitled, "Coding of Motion Vector Information," filed concurrently herewith.

3. Motion Compensation

A decoder uses a decoded motion vector to obtain a prediction macroblock (or field within a macroblock, etc.) in a reference frame. The horizontal and vertical motion vector components represent the displacement between the macroblock currently being decoded and the corresponding location in the reference frame. For example, positive values can represent locations that are below and to the right of the current location, while negative values can represent locations that are above and to the left of the current location.

If a current macroblock is frame-coded, one motion vector is used to obtain a prediction macroblock. In some embodiments, a decoder uses bi-cubic interpolation to obtain sub-pixel displacement. On the other hand, if the current macroblock is field-coded, the top field and bottom field have their own corresponding motion vectors. Accordingly, in some embodiments, given a field motion vector that points to a starting location in the reference frame, a decoder uses bi-cubic interpolation, taking alternating lines starting from the starting location, to compute the prediction field.

E. Interlaced B-frames

In some embodiments, a video encoder/decoder uses interlaced B-frames. For example, a video encoder/decoder encodes/decodes interlaced B-frames comprising macroblocks in a 4:1:1 format.

As explained above, in some embodiments, an encoder encodes macroblocks either as frame type or field type. For interlaced P-frames, an inter-coded field macroblock can have either one motion vector or two motion vectors. When an inter-coded field macroblock in a P-frame has two motion vectors, each of the two fields in the macroblock has its own motion vector and is compensated to form the residual. On the other hand, when an inter-coded field macroblock contains only one motion vector, one of the two fields is intra-coded while the other field is inter-coded.

In progressive B-frames, a macroblock can have from zero to two motion vectors, depending on the prediction mode for the macroblock. For example, in an encoder using five prediction modes (forward, backward, direct, interpolated and intra), forward and backward mode macroblocks have one motion vector for predicting motion from a previous reference or future frame. Direct mode macroblocks have zero motion vectors because in direct mode an encoder derives *implied* forward and backward pointing motion vectors – no actual motion vectors are sent for direct macroblocks. Intra mode macroblocks also have zero motion vectors. Interpolated mode macroblocks have two motion vectors (e.g., a backward motion vector and a forward motion vector).

For interlaced B-frames, an inter-coded field macroblock can have from zero to four motion vectors because each field can have from zero to two motion vectors, depending on the prediction mode of the field. For example:

- The encoder encodes no motion vector for the inter-coded field macroblock if both fields use direct or intra mode.
- The encoder encodes one motion vector if one field is either forward or backward predicted and the other field uses direct or intra mode.
- The encoder encodes two motion vectors if both fields use forward or backward prediction, or if the interpolated mode is used to predict one field and the other field uses direct or intra mode.
- The encoder encodes four motion vectors if both fields use the interpolated mode.

The set of possible motion vector combinations for a frame type B-frame macroblock is identical to the set of possible motion vector combinations for a progressive B-frame macroblock.

Although no motion vectors are sent for macroblocks that use direct mode prediction, direct mode macroblocks in interlaced frames are still designated as either frame type (using one motion vector for motion compensation) or field type (using two motion vectors for motion compensation), followed by the appropriate motion vector scaling and motion compensation in each case. This enables direct mode macroblocks in interlaced frames to be processed differently under different motion scenarios for better compression.

Having described and illustrated the principles of our invention with reference to various embodiments, it will be recognized that the various embodiments can be modified in arrangement and detail without departing from such principles. It should be understood that the programs, processes, or methods described herein are not related or limited to any particular type of computing environment, unless indicated otherwise.

Various types of general purpose or specialized computing environments may be used with or perform operations in accordance with the teachings described herein. Elements of embodiments shown in software may be implemented in hardware and vice versa.

In view of the many possible embodiments to which the principles of our invention may be applied, we claim as our invention all such embodiments as may come within the scope and spirit of the following claims and equivalents thereto.